# A Tool for Aligning Event Logs and Prescriptive Process Models through Automated Planning

Massimiliano de Leoni[1], Giacomo Lanciano[2], Andrea Marrella[2]

[1] Eindhoven University of Technology, The Netherlands
m.d.leoni@tue.nl
[2] Sapienza - University of Rome, Italy
lanciano.1487019@studenti.uniroma1.it,marrella@dis.uniroma1.it

**Abstract.** In Conformance Checking, *alignment* is the problem of detecting and repairing nonconformity between the actual execution of a business process, as recorded in an event log, and the model of the same process. Literature proposes solutions for the alignment problem that are implementations of planning algorithms built ad-hoc for the specific problem. Unfortunately, in the era of big data, these ad-hoc implementations do not scale sufficiently compared with well-established planning systems. In this paper, we tackle the above issue by presenting a tool, also available in ProM, to represent instances of the alignment problem as automated planning problems in PDDL (Planning Domain Definition Language) for which state-of-the-art planners can find a correct solution in a finite amount of time. If alignment problems are converted into planning problems, one can seamlessly update to the recent versions of the best performing automated planners, with advantages in term of versatility and customization. Furthermore, by employing several processes and event logs of different sizes, we show how our tool outperforms existing approaches of several order of magnitude and, in certain cases, carries out the task while existing approaches run out of memory.

## 1 Significance to the BPM field

Process mining is a discipline that sits between data mining and process modeling and analysis and, hence, can be considered one of the links between data science and process science [1]. The idea of process mining is to discover, monitor and improve the processes by extracting knowledge from the *event logs* that are stored in information systems about how these processes are executed within organizations.

Within process mining, *conformance checking* verifies whether the observed behavior stored in an event log is compliant with a process model that encodes how the process is allowed to be executed to ensure that norms and regulations are not violated. The notion of *alignment* [1, 2] provides a robust approach to conformance checking, which makes it possible to exactly pinpoint the deviations causing nonconformity with a high degree of detail. An alignment between a recorded process execution (*log trace*) and a process model is a pairwise matching between activities recorded in the log (*events*) and activities allowed by the model (*process activities*). Hence, in order to establish an alignment between a process model and an event log there is the need to relate "moves" in the log to "moves" in the model. However, it may be that some of the moves in the log cannot be mimicked by the model and vice versa, resulting in the presence of *deviations* between the model and the log.

In general, a large number of possible alignments exist between a process model and a log trace, since there may exist manifold explanations why a trace is not conforming. It is clear that one is interested in finding the most probable explanation, i.e., one of the alignments with the least expensive deviations (i.e., *optimal alignments* [1, 2]), according to some function assigning costs to deviations. The work by Adriansyah et al. [2] provides a technique to compute these optimal alignments, based on the A* algorithm and ad-hoc heuristics. However, experiments (also reported in this paper) show that their technique does not scale sufficiently. In the era of big data, scalable approaches are desperately necessary, as also advocated by the IEEE Task Force in Process Mining [3]: "*In some domains, mind-boggling quantities of events are recorded. [...] Therefore, additional efforts are needed to improve performance and scalability.*".

We believe that re-implementing standard planning algorithms, such as A*, for solving specific planning problems in an ad-hoc way is not ideal. First, ad-hoc implementations cannot compare in scalability with well-established planning systems. Second, ad-hoc implementations prevent one from seamlessly plugging in new algorithms or evaluate alternative heuristics. As a consequence, the possibility of evaluating different alternatives is hampered; also, if the literature proposes new algorithms that clearly overtake the existing ones for solving instances of the conformance checking problem, a massive amount of work is needed to modify those ad-hoc implementations.

In order to facilitate the integration of different planning algorithms, in this paper we present a tool that *(i)* allows to formulate the problem of computing optimal alignments as a *planning problem* in the standard Planning Domain Definition Language [4] (aka PDDL), and *(ii)* employs state-of-the-art automated planners [5] to solve such a problem. While the theoretical approach underlying the tool is discussed in [6], here we discuss its architecture and report on the most interesting results from the experiments, which show that our tool is versatile, allowing multiple planners to be integrated, and scales better than existing approaches with larger models and event logs.

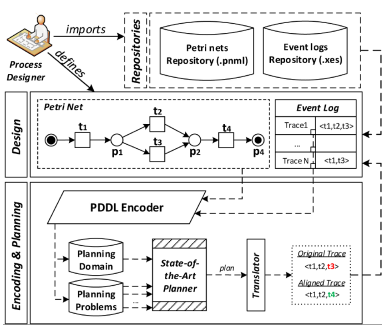
## 2 The Tool

The proposed tool is able to constructs an *optimal alignment* of an event log and a process model [1, 2]. To model business processes, we opted for Petri nets. The tool has been developed both as a plugin of ProM, a framework to develop process-mining techniques, and as a stand-alone standard Java application.[3] The ProM version is available in package *PlanningBasedAlignment* under a plugin named "*Planning-based Alignment of Event Logs and Petri Nets*". In the remainder, we mostly focus on the ProM version. The alignment engine relies on two main architectural layers as shown in Fig. 1(a).

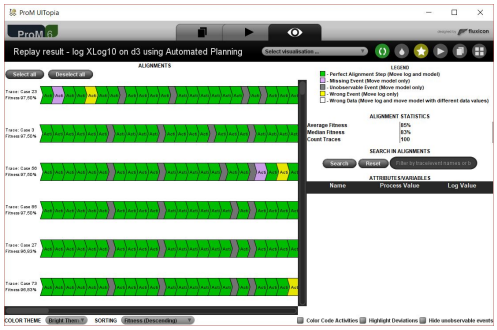
The *Design Layer* provides a wizard-based GUI that assists the process designer in: *(i)* the import of existing event logs and Petri nets stored in external repositories; *(ii)* the customization of a cost function on legal alignment moves to define the severity of a deviation[4]; *(iii)* the specification of the initial/final marking of the Petri net under analysis, and *(iv)* the selection of a search heuristic within an available repertoire. The tool supports the standards XES (eXtensible Event Stream) and PNML (Petri Net Markup Language) for respectively storing and analyzing event logs and Petri nets.

[3] The stand-alone version of the tool can be downloaded at `https://goo.gl/ZK9Frb`

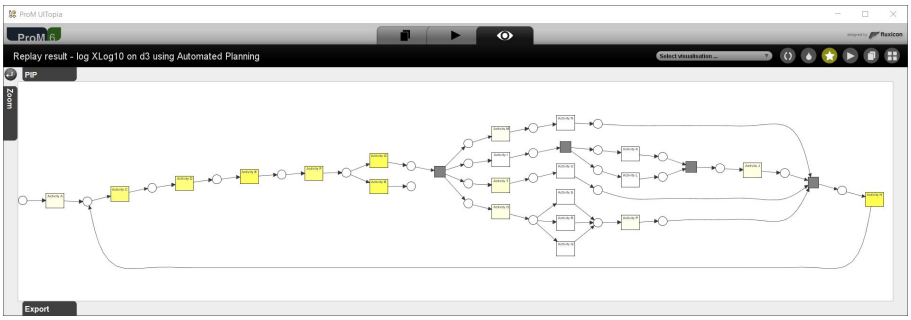
[4] The computation of the optimal alignment is based on this cost function: we want to minimize the cost of the alignment's moves.




(a) Architecture of the tool.




(b) Overview of the trace-model alignments.




(c) Projections of alignment moves on the Petri net.

**Fig. 1.** Architecture of the tool and some screenshots of the GUI as visualized in the ProM plugin.

The *Encoding & Planning Layer* is in charge of translating the input specifications in PDDL format, which is readable by any state-of-the-art planner, and performing the alignment task. Specifically, starting from a Petri net $N$ and an event log $L$ to be aligned, the *PDDL Encoder* builds, for each log trace $\sigma_L \in L$, *(i)* a PDDL domain file $P_D$, which encodes the domain propositions needed to capture the structure of $N$ and to monitor the evolution of its marking, and three classes of planning actions that represent "alignment" moves: synchronous moves (associated with no cost), model moves and log moves; and *(ii)* a PDDL problem file $P_R$, which includes a number of constants required to properly ground all the domain propositions in $P_D$; in our case, constants will correspond to the place and transition instances involved in $N$. Secondly, we define the initial state of $P_R$ to capture the exact structure of the specific log trace $\sigma_L$ of interest and the initial marking of $N$. Thirdly, we encode the goal condition of $P_R$ to represent the fact that $N$ is in the final marking and $\sigma_L$ has been completely analyzed. At this point, for any trace of the event log (i.e., for any of the generated planning problems) a state-of-the-art planner is invoked. With such inputs, the planner synthesizes a plan to reach the final goal from the initial state, i.e., *a sequence of alignment moves (each of which is a planning action) that establish an alignment between $\sigma_L$ and $N$*.

The ProM version is integrated with the FAST-DOWNWARD planning framework [7] to find optimal alignments of event logs and process models. FAST-DOWNWARD is a progression planner that uses hierarchical decompositions of planning tasks for com-

| | 10% noise | | | | 20% noise | | | | 30% noise | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Petri net Size | Existing Approach | Fast Downward | SymBA-2* | SPM&S | Existing Approach | Fast Downward | SymBA-2* | SPM&S | Existing Approach | Fast Downward | SymBA-2* | SPM&S |
| 25 | **25.85** | 84.79 | 685.35 | 812.44 | **42.8** | 86.21 | 692.97 | 814.04 | **56.87** | 88.87 | 699.42 | 816.05 |
| 36 | **11.71** | 94.13 | 767.05 | 950.25 | **19.24** | 92.78 | 785.14 | 956.19 | **28.6** | 92.67 | 793.13 | 959.25 |
| 68 | **112.85** | 130.68 | 1,413.85 | 1,722.13 | 164.92 | **134.57** | 1,418.51 | 1,779.17 | 278.31 | **144.99** | 1,419.5 | 1,808.61 |
| 95 | **77.56** | 123.65 | 2,048.84 | 2,431.35 | **119.89** | 125.78 | 2,049.14 | 2,445.66 | 168.32 | **126.82** | 2,053.92 | 2,483.98 |
| 115 | 638.52 | **205.47** | 2,289.84 | 2,787.31 | 966.34 | **384.26** | 2,291.91 | 2,817.29 | 1,987.12 | **716.68** | 2,352.81 | 2,868.77 |
| 136 | 304.62 | **178.66** | 2,752.28 | 3,282.23 | 465.97 | **182.88** | 2,765.14 | 3,300.87 | 578.63 | **187.17** | 2,775.46 | 3,317.48 |
| 175 | – | 18,759.91 | **5,909.46** | 6,546.87 | – | 84,195.41 | **6,092.93** | 6,837.57 | – | $1.71 \cdot 10^5$ | **6,259.99** | 7,182.13 |
| 263 | – | **2,343.92** | 11,960.84 | 13,289.56 | – | 13,524.63 | **12,044.67** | 13,744.92 | – | 30,582.11 | **12,194.09** | 14,075.09 |

**Table 1.** Comparison of the experimental results with different planners and with the existing approach of [2], using different models while varying the amount of noise. The values refers to the average time (in ms.) to compute the alignment of a log trace and a process model.

puting its heuristic function, called the *causal graph heuristic*, which approximates goal distances by solving a hierarchy of "local" planning problems. To produce optimal alignments, FAST-DOWNWARD uses a best-first search in first iteration to find a plan and a weighted A* search to iteratively decreasing weights of plans.

When a plan (i.e., an optimal alignment) satisfying the goal is found, the *Translator* component makes it available through the GUI. Fig. 1(b) shows the optimal alignments for the road-traffic management process and respective event log (see, e.g.,[8]).[5] Here, we leverage on the visualization developed for [8]: Each sequence of triangles refers to an optimal alignment of a trace with respect to the model. Each triangle is a different alignment move; the color of the move depends on its type: moves in the log, moves in the model or moves in both are colored in yellow, purple and green respectively. Each sequence is also associated with a number that identifies the *fitness score* of the specific trace. In addition, together with the alignment moves, when all the traces of a log have been analyzed, the tool produces several statistics to evaluate the quality of the alignments (e.g., the average cost of the alignment, the percentage of dirty traces in the log, etc.). Fig. 1(c) shows a screenshot of the tool where alignments are projected onto the process model. Each activity is associated with a color that reflects its degree of conformance, namely the fraction of log and model moves wrt. all moves for that activity. Red and white transitions indicate a degree 0 or 1 of conformance, respectively. Intermediate shades of red and yellow represent in-between values.

## 3 Maturity

Our tool was validated with several combinations of real-life and synthetic event logs and processes. We refer to [6] for a thorough discussions of the experimental test bed and of the results. Here, for the sake of space, we focus on the results on the synthetic process models and event logs. Specifically, we artificially generated eight process models of increasing sizes in form of Petri nets to evaluate the scalability of the approach and, for each Petri net, we automatically generated event logs (containing 1000 traces each) with increasing amount of noise. In the remainder, an event log with X% of noise indicates that, starting from an event log where each trace is conforming, an event is swapped with the next event in the trace with a probability of X%. This swapping generally causes the trace to be no more conforming.

[5] The log is available at `https://doi.org/10.1007/s00607-015-0441-1`

The experiments with the synthetic process models are summarized in Table 1. Each row refers to a different Petri net. For each row, the first column indicates the size of the Petri net in term of number of transitions; the subsequent 12 columns refer to the average time to alignment and event log trace with the respective process model when the corresponding event log contain 10%, 20% and 30% noise. For each amount of noise, we compared the result of the existing ad-hoc approach by Adriansyah et al. [2] with the results obtained by our tool. We ran our tool using its built-in planner, i.e., FAST-DOWNWARD, and two further external well-known state-of-the-art planners – SymBA-2* and SPM&S[6] – that provide searching algorithms that guarantee the optimality of the returned alignments. The results show that the existing ad-hoc approach is faster with models of small sizes and/or when using event logs with smaller amount of noise. Conversely, our approach is faster with larger models. In particular, it seems that, generally speaking, FAST-DOWNWARD performs better with middle-size models whereas SymBA-2* works better with very large models. The existing approach by Adriansyah et al. [2] is confirmed not to sufficiently scale when event logs and process models become very large. As a matter of fact, when testing with the two largest models, the existing approach was not able to align every log trace with the corresponding model and was running out of memory, even though we were using a i7 CPU machine with 16 GBs of RAM. Conversely, tracking the memory consumption, all of three employed planners were never using more than 4 GBs of RAM. Even when the existing approach was able to carry out the alignment tasks, it could do it significantly slower. As an example, compare the results for the model with 115 activities and an event log with 30%: On average, the existing approach requires 1987 ms per trace, versus 716 ms for FAST-DOWNWARD. A saving of 1.2 seconds per trace means that, to align all traces of an event log with, e.g., 100000 traces (not uncommon in real-world case studies), the use of FAST-DOWNWARD would allow us to save 120000 seconds: 83.3 hours.

**Acknowledgments.** The work of Andrea Marrella has been partly supported by the Sapienza project "Data-aware Adaptation of Knowledge-intensive Processes in Cyber-Physical Domains through Action-based Languages".

## References

1. van der Aalst, W.M.P.: Process Mining: Data Science in Action. Springer Berlin (2016)
2. Adriansyah, A., Sidorova, N., van Dongen, B.F.: Cost-Based Fitness in Conformance Checking. In: 11th Int. Conf. on Application of Concurrency to System Design, IEEE (2011)
3. Van Der Aalst, W., et al.: Process Mining Manifesto. In: 9th Int. Conf. on Business Process Management (BPM 2011), Springer Berlin (2011)
4. McDermott, D., et al.: PDDL—The Planning Domain Definition Language. Technical Report DCS TR-1165, Yale Center for Computational Vision and Control (1998)
5. Geffner, H., Bonet, B.: A Concise Introduction to Models and Methods for Automated Planning. Synthesis Lectures on Artificial Intelligence and Machine Learning **8**(1) (2013)
6. de Leoni, M., Marrella, A.: Aligning Real Process Executions and Prescriptive Process Models through Automated Planning. Expert System with Applications **82** (2017)
7. Helmert, M.: The Fast Downward Planning System. JAIR **26** (2006)
8. Mannhardt, F., de Leoni, M., Reijers, H.A., van der Aalst, W.M.P.: Balanced multi-perspective checking of process conformance. Computing **98**(4) (2015)

[6] Planners are downloadable at: `https://helios.hud.ac.uk/scommv/IPC-14/`